C of C
TFW

# *IN THE UNITED STATES PATENT AND TRADEMARK OFFICE*

In re application of: Slaughter et al.

Attorney Docket No.: SUN1P209/P3661

Patent No.: 6,862,735 B1

Issued: March 1, 2005

Title: MECHANISM BY WHICH PLATFORM INDEPENDENT SOFTWARE MAY BIND TO AND ACCESS PLATFORM DEPENDENT SOFTWARE

09,249,229

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as first-class mail on April 8, 2005 in an envelope addressed to the Commissioner for Patents, P.O. Box 1450 Alexandria, VA 22313-1450.

Signed: _____
Aurelia M. Sanchez

## REQUEST FOR CERTIFICATE OF CORRECTION OF OFFICE MISTAKE (35 U.S.C. §254, 37 CFR §1.322)

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450
Attn: Certificate of Correction

Dear Sir:

Attached is Form PTO-1050 (Certificate of Correction) at least one copy of which is suitable for printing. The errors together with the exact page and line number where the errors are shown correctly in the application file are as follows:

**SPECIFICATION:**

1. Column 3, line 42, change "the method" to --the platform dependent method--. This appears correctly in the patent application as filed on February 11, 1999, on page 5, line 24.

**CLAIMS:**

1. In line 4 of claim 1 (column 7, line 5) change "platform dependent" to --platform independent--. This appears correctly in Amendment D as filed on July 22, 2004, on page 2, paragraph 1, line 3.

2. In line 3 of claim 2 (column 7, line 12) change "platform dependent" to --platform independent--. This appears correctly in Amendment D as filed on July 22, 2004, on page 2, paragraph 2, line 3.

3. In line 1 of claim 9 (column 7, line 34) change "clam 8" to --claim 8--. This appears correctly in Amendment D as filed on July 22, 2004, on page 3, paragraph 2, line 1.

4. In line 2 of claim 16 (column 8, line 6) change "plat independent" to --platform independent--. This appears correctly in Amendment D as filed on July 22, 2004, on page 4, paragraph 3, line 2.

5. In line 3 of claim 17 (column 8, line 11) change "compute system" to --computer system--. This appears correctly in Amendment D as filed on July 22, 2004, on page 4, paragraph 4, line 2.

6. In line 9 of claim 21 (column 8, line 54) change "to platform" to --the platform--. This appears correctly in Amendment D as filed on July 22, 2004, on page 5, paragraph 4, line 8.

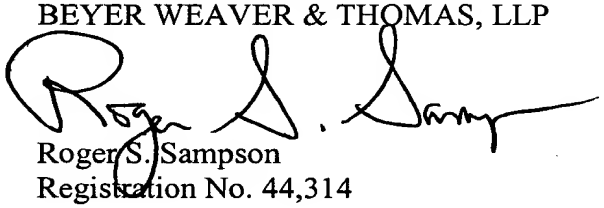
7. In line 1 of claim 22 (column 8, line 59) change "for transmitting" to --for transforming--. This appears correctly in Amendment D as filed on July 22, 2004, on page 5, paragraph 5, line 1.

8. In line 1 of claim 23 (column 9, line 7) change "on encapsulating" to --encapsulation--. This appears correctly in Amendment D as filed on July 22, 2004, on page 6, paragraph 1, line 1.

Patentee hereby requests expedited issuance of the Certificate of Correction because the error lies with the Office and because the error is clearly disclosed in the records of the Office. As required for expedited issuance, enclosed is documentation that unequivocally supports the patentee's assertion without needing reference to the patent file wrapper.

It is noted that the above-identified errors were printing errors that apparently occurred during the printing process. Accordingly, it is believed that no fees are due in connection with the filing of this Request for Certificate of Correction. However, if it is determined that any fees are due, the Commissioner is hereby authorized to charge such fees to Deposit Account 500388 (Order No. SUN1P209).

Respectfully submitted,
BEYER WEAVER & THOMAS, LLP



Roger S. Sampson
Registration No. 44,314

P.O. Box 70250
Oakland, CA 94612-0250
650-961-8300

## DESCRIPTION OF THE PREFERRED EMBODIMENT

The present invention includes software, methods, and apparatus that are arranged to permit platform-independent objects to access platform dependent software. In one described embodiment of the present invention, an encapsulation object that includes a wrapper arranged to call an associated platform independent method is described. In the described embodiment, substantially the only operation performed by the wrapper is to act as an intermediary between a platform independent object and the platform dependent method to facilitate calling the platform dependent method from the platform independent service.

Figure 1 illustrates a native encapsulation object 100 in accordance with an embodiment of the invention. The native encapsulation object 100 includes wrappers 102 and corresponding platform dependent methods (also referred to as native methods) 104. In one embodiment of the invention, each of the wrappers 102 is arranged to encapsulate (or call) an associated one of the platform dependent methods 104. In this way, a platform independent object, such as for example, a device driver, can access platform dependent software (represented by the native methods) using a platform independent protocol by first calling the wrappers 102 associated with the platform independent software. The wrapper, in turn, calls (or invokes) the appropriate platform independent software, which is then accessible to the platform independent object.

In one particular embodiment, some of the wrappers 102 take the form of Java wrapper 102a through 102n that are well known to those skilled in the art. By way of example, the Java wrapper 102a is associated with a platform dependent method 104a included in the methods 104. A platform independent object 106, such as a device driver, can preserve its platform independence by first calling the Java wrapper 102a instead of directly calling the native method 104a. The Java wrapper 102a responds by directly calling the platform dependent method 104a. The method 104a is then accessible to the platform independent object 106. Since the object 106 has only called the Java wrapper 102a (which is by definition platform

## LISTING OF CLAIMS:

1. (currently amended) A software object included in a computer system, comprising:

a platform dependent method; and

a platform independent wrapper arranged to encapsulate and to call the platform dependent method, wherein a platform independent object accesses the platform dependent method by calling the wrapper, wherein the wrapper then calls the platform dependent method.

2. (original) A software object as recited in claim 1, wherein substantially the only operation performed by the wrapper is to act as an intermediary between the platform independent object and the native method to facilitate calling the platform dependent native method from the platform independent object.

3. (original) A software object as recited in claim 2, wherein the software object is one of a plurality of software objects included in the computer system.

4. (original) A software object as recited in claim 3, wherein the platform dependent method is one of a plurality of platform dependent methods.

5. (original) A software object as recited in claim 4, wherein the wrapper is one of a plurality of wrappers each being arranged to call an associated one of the plurality of platform dependent methods.

6. (original) A software object as recited in claim 5, wherein a first software object includes a first wrapper and an associated first method designed to run on a first platform.

7. (original) A software object as recited in claim 6, wherein a second software object includes a second wrapper and an associated second method designed to run on a second platform that is different than the first platform.

8. (original) A software object as recited in claim 7, wherein the wrapper is a Java wrapper.

9. (original) A software object as recited in claim 8, wherein the platform independent object is a Java device driver.

10. (currently amended) A computer-implemented method of accessing a platform dependent method by a platform independent object in a computer system, the computer system having an encapsulation object, the method comprising:

calling a platform independent wrapper by the platform independent object; and

calling the method by the wrapper wherein the wrapper encapsulates the method and the method is included in the encapsulation object.

11. (original) A computer implemented method as recited in claim 10, wherein substantially the only operation performed by the wrapper is to act as an intermediary between the platform independent object and the method so as to facilitate calling the platform dependent method from the platform independent object.

12. (original) A computer implemented method as recited in claim 10, wherein the encapsulation object is one of a plurality of encapsulation objects included in the computer system, and wherein the wrapper is one of a plurality of wrappers included in the computer system, and wherein the method is one of a plurality of methods included in the computer system.

13. (original) A computer implemented method as recited in claim 12, wherein the computer system includes a first encapsulation object containing a first wrapper and an associated first method, wherein the first method is designed to run on a first platform.

14. (original) A computer implemented method as recited in claim 13, wherein the computer system includes a second encapsulation object containing a second wrapper and an associated second method that is designed to run on a second platform that is different than the first platform.

15. (original) A computer implemented method as recited in claim 14, wherein the platform independent object accesses the first method by calling the first wrapper that, in turn, calls the first method.

16. (original) A computer implemented method as recited in claim 14, wherein the platform independent object accesses the second method by calling the second wrapper that, in turn, calls the second method.

17. (previously presented) A computer-implemented method of accessing a platform dependent method by a platform independent object in a computer system, the computer system including a system manager, an encapsulation object that contains the platform dependent method and a business card associated with the platform independent object, the business card containing configuration data that includes an encapsulation object pointer that is used to identify the encapsulation object containing the platform dependent method, comprising:

requesting the encapsulation object containing the platform dependent method from the system manager by the platform independent object;

retrieving the business card corresponding to the requesting object by the system manager;

retrieving the encapsulation object pointer from the requesting object's business card by the system manager;

instantiating the encapsulation object identified by the encapsulation object pointer;

calling a platform independent wrapper contained in the encapsulation object corresponding to the platform dependent method by the platform independent object; and

calling the platform dependent method by the wrapper.

18. (previously presented) The method as recited in claim 17, wherein the business card is instantiated by a system administrator at system startup.

19. (original) The method as recited in claim 18, wherein the platform independent object is a device driver, wherein the device driver is used to manage a device coupled to the computer system.

20. (original) The method as recited in claim 19, wherein the system manager is a bus manager used to manage a bus coupled to the device.

21. (original) A computer implemented method of transforming a mixed software object containing platform independent code and platform dependent code into a pure software object having only platform independent code, comprising:

instantiating an encapsulating object containing all the platform dependent code, the encapsulating object including a wrapper associated with and used to invoke the platform dependent code;

removing all the platform dependent code from the mixed software object; and

replacing all calls to the platform dependent code in the mixed software object with calls to the wrapper included in the native encapsulating object.

22. (original) A computer program product for transforming a mixed software object containing platform independent code and platform dependent code into a pure software object having platform independent code only, the computer program product comprising:

computer code that writes an encapsulating object containing all the platform dependent code, the encapsulating object including a Java wrapper used to invoke the platform dependent code;

computer code that removes all the platform dependent code from the mixed software object; and

computer code that replaces all calls to the platform dependent code in the mixed software object with calls to the wrapper included in the native encapsulating object.

23. (currently amended) A native encapsulation object included in a computer system, the native encapsulation object comprising:

a plurality of device drivers; and

a plurality of wrappers arranged to encapsulate and to call an associated device driver of the plurality of device drivers, wherein a platform independent object accesses an associated device driver by calling one of the plurality of wrappers, wherein the wrapper then calls the associated device driver.

PTO/SB/44 (10-96)
Approved for use through 6/30/99. OMB 0651-0033
Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE


(Also Form PT-1050)

# UNITED STATES PATENT AND TRADEMARK OFFICE
# CERTIFICATE OF CORRECTION

PATENT NO. : 6,862,735 B1

DATED : March 1, 2005

INVENTOR(S) : Slaughter et al.

It is certified that error appears in the above-identified patent and that said Letters Patent are hereby corrected as shown below:

**In the Specifications:**

Column 3, line 42, change "the method" to --the platform dependent method--.

**In the Claims:**

In line 4 of claim 1 (column 7, line 5) change "platform dependent" to --platform independent--.

In line 3 of claim 2 (column 7, line 12) change "platform dependent" to --platform independent--.

In line 1 of claim 9 (column 7, line 34) change "clam 8" to --claim 8--.

In line 2 of claim 16 (column 8, line 6) change "plat independent" to --platform independent--.

In line 3 of claim 17 (column 8, line 11) change "compute system" to --computer system--.

In line 9 of claim 21 (column 8, line 54) change "to platform" to --the platform--.

In line 1 of claim 22 (column 8, line 59) change "for transmitting" to --for transforming--.

In line 1 of claim 23 (column 9, line 7) change "on encapsulating" to --encapsulation--.

MAILING ADDRESS OF SENDER:

PATENT NO. 6,862,735 B1

**Roger S. Sampson**
BEYER WEAVER & THOMAS, LLP
P.O. Box 70250
Oakland, CA 94612-0250

No. of Additional Copies

→ 1